Deliverable

Grant Agreement Number	270137
Full Project Title	Scalable Preservation Environments
Project Acronym	**SCAPE**
Title of Deliverable	Knowledge base of format-specific preservation risks: the OPF File Format Risk Registry
Deliverable Number	D11.4
Work-package	WP11
Dissemination Level	
Deliverable Nature	**O** = Other
Contractual Delivery Date	2014-07-31
Actual Delivery Date	2014-07-22
Author(s)	KB

Abstract

This report gives an overview of the work that was done as part of the preservation risks knowledge base task. This includes the establishment of the OPF File Format Risk Registry, research and development on analysis tools, the creation of evidence in the form of sample files, and the development of a general methodology for assessing files against an institutional policy. The relations of this work with SCAPE activities on preservation watch, policies and quality assurance are also discussed.

Keyword list

preservation risks, threats, file formats, detection, tools, assessment, corpora, pdf, jp2, jpeg 2000.


SCAPE

Authors

Person	Role	Partner	Contribution
Johan van der Knijff	Author	KB	Main author

Document Approval

Person	Role	Partner
Luis Faria	Reviewer	KEEPS

Distribution

Person	Role	Partner
Luis Faria	Reviewer	KEEPS
Ross King	Coordinator	AIT

Revision History

Version	Status	Author	Date	Changes
0.5	Draft	Johan van der Knijff	2014-05-27	Initial document draft
0.6	Draft	Johan van der Knijff	2014-07-14	Revision work
0.7	Draft	Johan van der Knijff	2014-07-15	Added section 3.3
1.0	Final	Johan van der Knijff	2014-07-21	Added clarification on discrepancy between task description and deliverable description
1.1	Final	Johan van der Knijff	2014-07-22	Fixed formatting problem

References

Ref.	Document	Date	Details and Version
D12.2	Final version of the Preservation Watch component	2012-01-27	
D13.1	Policy specification model	2013-07-31	
D11.2	Quality Assurance Workflow, Release 2 + Release Report	2013-05-31	


SCAPE

Executive Summary

Many file formats have features that are not compatible with the aims of long-term preservation. For example, the use of encryption may result in content becoming inaccessible over time, and files that have dependencies on external resources (e.g. fonts that are not embedded) may not be rendered correctly. Such features are potential preservation risks, and for digital archives it is important to be aware of the fact what risks are associated with a specific format, and how files can be assessed for the presence of a risk. The overall aim of this deliverable is to address this situation by presenting an open knowledge base of format-specific preservation risks.

This was realised by creating a platform that is called the OPF File Format Risk Registry (FFRR), which is part of the Open Planets Knowledge Base Wiki. For a given format, FFRR lists the risks that are associated with it, it provides information on how to test if a file is affected a risk, and it gives recommendations on how to deal with file objects that are affected by a risk. It also provides evidence in the form of sample files. FFRR content was created for the formats JP2 (JPEG 2000 Part 1) and PDF. These formats were chosen because of their particular importance to other SCAPE partners, and they are used here as a proof of concept of the FFRR approach. Research was done on tools that are able to detect preservation risks for these formats. The results of this research were included in FFRR. As no suitable tool existed for JP2, a new tool was developed for this format. As file format risks may be explicitly addressed by institutional policies, a general methodology was developed for an automated assessment (or validation) of files against a policy.

Please note that the description of the task that led to this deliverable has been changed through the project's change procedure. However, the deliverable description was not adapted accordingly. This deliverable follows the new task description.

Table of Contents

1 Introduction 9
1.1 Overall scope of this work 9
1.2 Requirements 10
1.3 Specific goals 11
1.4 Outline of this report 12
2 Proof of concept knowledge base 13
2.1 Platform: the OPF File Format Risk Registry 13
2.2 JP2 (JPEG 2000 Part 1) 14
2.2.1 Jpylyzer development and relation to FFRR work 15
2.2.2 Assessment 15
2.2.3 Evidence 16
2.2.4 Recommendations 16
2.3 PDF 16
2.3.1 Assessment 17
2.3.2 Evidence 17
2.3.3 Recommendations 18
3 Policy-based assessment 19
3.1 General methodology 19
3.2 Policy-based assessment of PDF 21
3.3 Concluding remarks 23
4 Relations with other activities 25
4.1 Preservation watch and SCOUT 25
4.2 Policies 26
4.3 Quality Assurance 26
4.4 Existing format registries 26
4.5 Uptake of results by industry and community 27
4.5.1 Jpylyzer 27
4.5.2 Apache Preflight 27

5 Conclusions 29
Annex: Products and publications 31
OPF File Format Risk Registry 31
jpylyzer software 31
Corpora and data 31
Publications on jpylyzer and JPEG 2000 31
Publications on PDF 32
Miscellaneous publications 32


SCAPE

1 Introduction

Many file formats have features that are not compatible with the aims of long-term preservation. To name but a few examples:

- The PDF format makes it possible to encrypt the contents of a document. If the decryption key or password of an encrypted document gets lost, it may not be (legally) accessible in the future.
- The rendering of a PDF document that uses fonts that are not embedded in the file may be very different from its original "look and feel" (depending on the viewing environment).
- JPEG 2000 images created by some encoder applications contain colour space information that is embedded in a non-standard manner, which can result in interoperability problems.

Features like these are potential preservation risks, and dealing with them can be challenging for digital archives. First of all, they have to be aware of the risks that are associated with a specific format. Subsequently they will want to know if a format instance (i.e. a single file object) is affected by any of these risks. How does one assess or measure this? Which tools are available, and how should one interpret a tool's output? Finally, if the assessment shows that a file object is affected by a specific risk, this may trigger some action, for example:

- reject it from being ingested into the repository (ingest workflow);
- record the presence of the risk and its implications in the metadata;
- fix/normalise the file so that the risk goes away.

1.1 Overall scope of this work

The overall aim of this deliverable is to develop a proof of concept of an open knowledge base of format-specific preservation risks. It is targeted at an audience of digital preservation professionals, and its contents are meant to be “consumed” by humans (i.e. machine-readability is out of scope, although in some cases it may be possible to translate content into something that is machine-readable). Apart from the creation of a *platform* for recording preservation risks, a major effort was made to create *content*, and to demonstrate how this content can be used to detect preservation risks in actual collections.

Early on in the project it was decided to restrict the content creation for the proof of concept to the following two formats:

- The JP2 (JPEG 2000 Part 1) still image format;
- Portable Document Format (PDF).

The principal reason for selecting these two specific formats is that they are of special importance to a large number of partners in the SCAPE project. JP2 is rapidly gaining importance as both a master and access image format in many large-scale digitisation projects. PDF is mainly important because of its ubiquity, combined with the fact that it is a highly complex format that contains various features that are problematic from a preservation point of view. Moreover, the SCAPE Description of Work explicitly states that the knowledge base efforts should “provide input to the JPEG 2000 and PDF-related QA work” in SCAPE Work Package 11.

1.2 Requirements

As mentioned before, for digital archives to deal with format-specific preservation risks in an effective manner, they will need to be aware of the risks, and they will need to know how to test (measure) if a file object is affected by them. This implies that a knowledge base on file format risks must meet the following minimum requirements to be of any practical value:

1. It must clearly *describe*/identify each risk.
2. It must provide *evidence* of each risk (sample files).
3. It must provide information on how we can test/measure if a file object is affected by each risk (*assessment*).

Figure 1 below gives a graphical representation of this . It shows a file format that has a number of associated preservation risks. Each risk has a *description*, which simply describes the offending feature and explains why it is a preservation risk. As an example, we may think of the use of non-embedded fonts in a PDF document. This feature can be a preservation risk, as it can result in faulty rendering of documents. The *assessment* component describes how a PDF can be tested for non-embedded fonts. It does this by listing one or more software tools that are capable of doing this. It also describes how the output of the tool should be interpreted (e.g. by listing the output elements that uniquely identify the presence of non-embedded fonts). The *evidence* for this risk will then be one or more PDF documents that have non-embedded fonts. This serves a number of purposes:

- It makes the claims in the description verifiable.
- It will allow users of the knowledgebase to verify the results of the assessment procedure.
- It will allow (external) contributors to improve the assessment procedure (or to come up with alternative procedures).

In some cases it may be possible to provide recommendations on how to deal with file objects that are affected by a risk (e.g. mitigate its impact). So, this should be covered by the knowledge base as well. A final requirement is that the knowledge base must be *open*, as this will allow external contributors to add new content and to improve existing content.

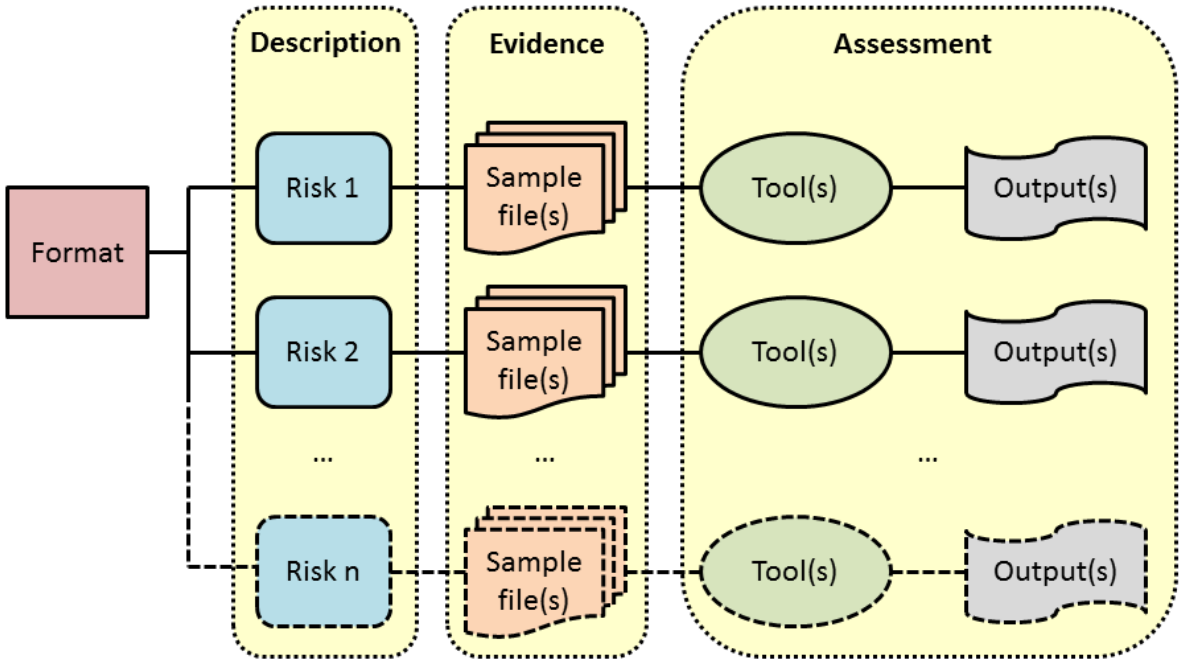

Figure 1 *Formats, risks, evidence and assessment*

1.3 Specific goals

The goals of the work covered by this deliverable were to:

1. Create a platform for a knowledge base on format-related preservation risks.
2. Fill the knowledge base for the formats JP2 and PDF, focusing on the description of preservation risks and their assessment in file objects. This should result in entries that are of direct use to content holders that are working with these formats. This also provides input to other SCAPE activities on these formats (see chapter 4), and serves as a proof of concept for other formats that may be added to the knowledge base after the end of SCAPE.
3. Do research on tools that can be employed to detect preservation risks for the proof of concept formats, and incorporate the results into the knowledge base.
4. Develop a general methodology to validate the presence of preservation risks in file instances against an institutional collection policy (policy-based assessment) .

At an early stage of the project it became obvious that no suitable assessment tool was available for the JP2 format, which lead to the development of an entirely new tool for this format (jpylyzer). This development task was carried out as part of the work covered by this deliverable, since the availability of such a tool is absolutely essential for the detection of preservation risks for this format. Evidence in the form of openly licensed sample files turned out to be lacking as well, so the work also comprised the creation of a some test corpora to fill this gap. The result is that to realise the aforementioned goals, the work that is covered by this deliverable followed a number of different strands. The main purpose of this report is to provide an overview of these strands, to show how they relate to each other, and to explain

how the work as a whole is related to other SCAPE activities on preservation watch, policies and quality assurance.

1.4 Outline of this report

Chapter 2 gives an overview of the proof of concept of the knowledge base. It describes the platform, and the work that was done on the JP2 and PDF formats. This includes the research and development that was done on tools, and the creation of supporting evidence in the form of sample files. Chapter 3 presents a general methodology for assessing file objects against an institutional policy, where the policy is made up of objectives that each relate to a specific preservation risk. Chapter 4 puts this work into context by outlining its relations with other activities within (and outside of) SCAPE. Chapter 5 summarises the main conclusions. An Annex lists (and provides links to) all products and publications that were created as part of this work.

2 Proof of concept knowledge base

This chapter gives an overview of the proof of concept of the knowledge base. The first section describes the platform, followed by two sections that cover the work that was done on the JP2 and PDF formats. This includes the research and development that was done on tools, and the creation of supporting evidence in the form of sample files.

2.1 Platform: the OPF File Format Risk Registry

After an initial appraisal of the nature of the information that would be part of the knowledge base, it was decided to implement it as a simple Wiki-based platform. This also makes the process of contributing and editing content simple for external contributors, which is important for an open knowledge base. Since an *Open Planets Foundation's knowledge base wiki* already existed[1], the knowledge base on file format risks was created as a set of pages within that wiki [2]. The SCAPE project also includes another "knowledge base", which is a component of the SCOUT preservation watch system (see chapter 4). To avoid any confusion between these two, the "knowledge base" that is covered by this deliverable was given the moniker *OPF File Format Risk Registry* (hereafter: FFRR).

Figure 2 shows the entry page of FFRR. Its main building blocks are *format* pages and *format issue* pages. Although the underlying Wiki platform doesn't dictate how the information in either is structured, templates were created to provide contributors some guidance and to encourage a common general format.

For the duration of SCAPE, content was created for two formats only: JP2 and PDF. The reasons for this were given in section 1.1. The following sections outline the work done on these formats. Since the FFRR is completely open, external contributors are free to add entries on other formats (and several external contributions exist already). The KB also envisages additional entries on the EPUB format, but work on this will be done outside of SCAPE, and after the end of the project .

[1] http://wiki.opf-labs.org

[2] http://wiki.opf-labs.org/display/TR/OPF+File+Format+Risk+Registry

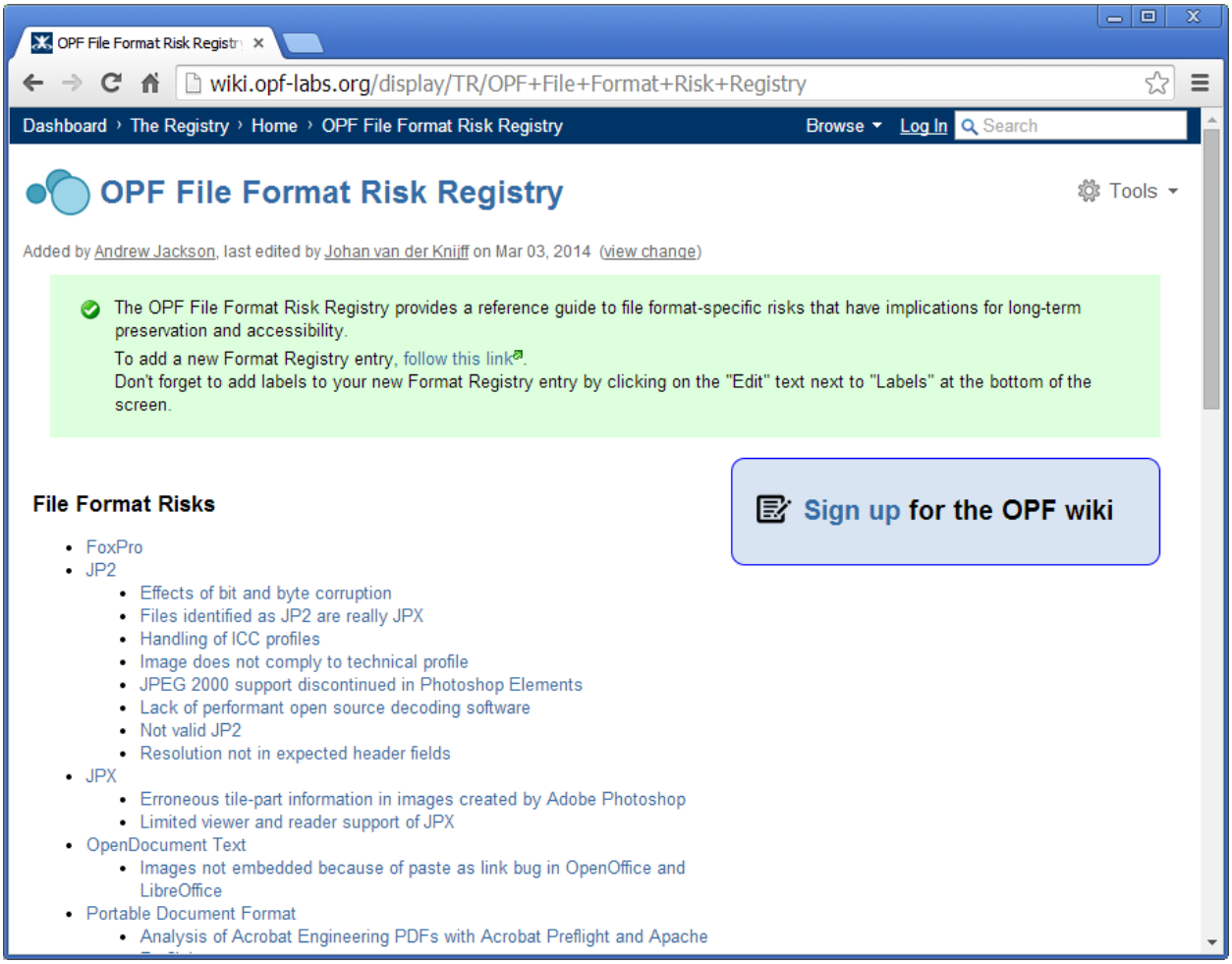

Figure 2 *Entry page of the OPF File Format Risk Registry (FFRR).*

2.2 JP2 (JPEG 2000 Part 1)

JP2 is the still image format that is defined by Part 1 of the JPEG 2000 standard[3]. Over the last few years the adoption of this format has seen a steep rise in the cultural heritage sector, and many institutions (including SCAPE partners the National Library of the Netherlands, the Austrian National Library, the British Library and the Danish State and University Library) are now using JP2 as a preferred archival and/or access format for digital imagery. The use of JP2 introduces several risks, most of which are the result of either one of the following general problems:

- ambiguities in (previous versions of) the format specification;
- software implementations that don't follow the standard;
- lack of (performant) analysis tools, which can make quality assurance problematic.

[3] http://www.jpeg.org/public/15444-1annexi.pdf

These are all further specified in the JP2 entry in FFRR[4]. The issues listed here are largely based on previous work by the National Library of the Netherlands[5], and experiences of byte-corrupted images by SCAPE partner The British Library[6].

2.2.1 Jpylyzer development and relation to FFRR work

Soon after the project's start it became apparent the assessment of JP2s for specific preservation risks (esp. damaged or otherwise invalid JP2s) was highly problematic using existing tools. For instance, it turned out that the JPEG 2000 module of the widely-used JHOVE[7] tool was unable to detect truncated and otherwise incomplete JP2 images. Various other deviations from the format's specification went unnoticed by JHOVE as well. This prompted the development of a simple JP2 file structure checker[8], which was soon after expanded to jpylyzer, a full-fledged validator and properties extractor[9]. Jpylyzer was developed as part of this Knowledge Base work. It has its own website[10], and is fully documented by an exhaustive User Manual, which contains detailed information on jpylyzer's installation procedure, its usage and its reported output. Moreover, it includes descriptions of every single test that is performed for validation, and every reported property[11].

2.2.2 Assessment

The *Assessment* section of each *Issue* entry in FFRR describes how a specific issue/risk can be identified in a file (if applicable). For example, one entry describes the issue of ambiguous information in the JP2 format specification about the embedding of ICC profiles, which can lead to a number of interoperability problems[12].The *Assessment* section of this entry tells one which tool to use (jpylyzer), and how to establish if a file is affected from its output. In this case, the most obvious result is that affected files fail the validation. FFRR provides information on what jpylyzer's output will look like in that case (represented as an XPath expression):

Tool	**Affected if expression returns True**
Jpylyzer	`"/jpylyzer/isValidJP2 = 'False'"`

[4] http://wiki.opf-labs.org/display/TR/JP2

[5] http://dlib.org/dlib/may11/vanderknijff/05vanderknijff.html; http://jpeg2000wellcomelibrary.blogspot.nl/2010/12/guest-post-ensuring-suitability-of-jpeg.html

[6] http://wiki.opf-labs.org/display/REQ/Truncated+JPEG2000

[7] http://jhove.sourceforge.net/

[8] http://www.openplanetsfoundation.org/blogs/2011-09-01-simple-jp2-file-structure-checker

[9] http://www.openplanetsfoundation.org/blogs/2011-12-14-prototype-jp2-validator-and-properties-extractor

[10] http://openplanets.github.io/jpylyzer/

[11] http://openplanets.github.io/jpylyzer/userManual.htm

[12] http://wiki.opf-labs.org/display/TR/Handling+of+ICC+profiles

It also provides additional information on the most important reported properties that are affected by this issue, as well as an overview of how the most widely-used JPEG 2000 implementations handle ICC profiles.

2.2.3 Evidence

Entries on preservation risks are supported by evidence in the form of sample files where possible. For example, the entry on the ICC profiles contains links to two sample files that are affected by this issue (section *Example files*). For this purpose an annotated set of openly-licensed sample images was created[13], which is part of the Open Planets Format Corpus[14].

2.2.4 Recommendations

Where possible, FFRR provides general recommendations on how to deal with file objects that are affected by specific risks. Taking the entry on ICC profiles as an example, one could document the presence of problematic ICC profiles in the metadata, or normalise affected files to valid JP2.

2.3 PDF

The Portable Document Format (PDF) is intended to provide a platform-independent representation of formatted documents. It has its origins in (and is based on) the PostScript page description language. For preservation the most relevant aspects of the format are:

1. it is ubiquitous;
2. it is highly complex and rich in features;
3. it includes various features that are at odds with long-term accessibility.

The issues and risks that are listed in FFRR's PDF entry[15] are largely based on previous work by the National Library of the Netherlands[16] (which was the result of a review of the format's specification), the specification of PDF/A-1[17], and input from other memory institutions[18]. There is a focus on the following issues, which present the most urgent preservation risks:

- non-conformance to the format specification;
- encryption;
- font issues (including non-embedded, incomplete and damaged fonts);
- multimedia content;
- JavaScript;
- file attachments;

---

[13] https://github.com/openplanets/format-corpus/tree/master/jp2k-test

[14] https://github.com/openplanets/format-corpus

[15] http://wiki.opf-labs.org/display/TR/Portable+Document+Format

[16] http://www.openplanetsfoundation.org/system/files/PDFInventoryPreservationRisks_0_2_0.pdf

[17] http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=38920

[18] E.g. some of the discussions here:
https://web.archive.org/web/20130514095533/http://libraries.stackexchange.com/questions/tagged/pdf

- external dependencies.

2.3.1 Assessment

Since the PDF/A standard already prohibits the use of most features that are problematic for long-term preservation, an investigation was started to test if existing PDF/A validation tools could be used to detect specific risks in *any* given PDF document (including PDFs that don't follow the PDF/A standard at all). This approach turned out to work quite well using Apache Preflight, an open-source PDF/A-1 validator that is part of the PDFBox[19] library[20,21]. The investigation did reveal a number of shortcomings of Preflight; however the most serious of these were fixed by its developers soon after reporting them.

The entry on multimedia[22] illustrates how FFRR can help a content holder to assess a PDF for the presence of various types of multimedia content (which can be a preservation risk) with Apache Preflight. Here, FFRR lists the specific validation errors that Preflight reports whenever it encounters multimedia content. Preflight always reports validation errors using a combination of an error code and a detailed description of the error. In many cases the error code by itself is not specific enough. For example, a PDF with multimedia content may result in error 5.2.1, which is a generic code for a "forbidden field in an annotation definition". This is not very informative, and in fact it may be reported for files that do not contain multimedia content at all. It is only the detailed error description that specifies the exact *type* of the field that isn't allowed (e.g. a "Screen" or "Movie" annotation). So, the role of FFRR here is to provide exactly this information. It should be noted that the PDF entries are generally more complex than those on JP2. One reason for this is that PDF is simply a more complex format. Many of its features can be implemented in different ways, where each may result in a different validation error. The manner in which validation errors are reported by Preflight also complicates things somewhat: many of the error codes are generic, which means that specific combinations of error codes and detailed error descriptions need to be considered to accurately identify some of the preservation risks[23].

2.3.2 Evidence

Just like JP2, an annotated set of openly-licensed sample files was created to provide evidence of each risk[24]. These are all small files that were specifically created to showcase one single risk. As PDFs "in the wild" turned out to result in more heterogeneous output from Preflight, additional tests[25] were done

[19] https://pdfbox.apache.org/

[20] http://www.openplanetsfoundation.org/blogs/2012-12-19-identification-pdf-preservation-risks-apache-preflight-first-impression

[21] http://www.openplanetsfoundation.org/blogs/2013-07-25-identification-pdf-preservation-risks-sequel

[22] http://wiki.opf-labs.org/display/TR/Multimedia+content

[23] For convenience FFRR includes the following summary Preflight errors: http://wiki.opf-labs.org/display/TR/Summary+of+Apache+Preflight+errors

[24] https://github.com/openplanets/format-corpus/tree/master/pdfCabinetOfHorrors

[25] http://wiki.opf-labs.org/display/TR/Analysis+of+Acrobat+Engineering+PDFs+with+Acrobat+Preflight+and+Apache+Preflight

using files taken from Adobe's Acrobat Engineering website[26]. This made it possible to produce a more complete picture of the errors reported by Preflight for each preservation risk. Note that the files on the Acrobat Engineering website are not openly licensed, which means that they couldn't be included or re-used as part of FFRR. Since this dataset captures many advanced and less commonly used features of PDF, it was decided to refer to it in FFRR anyway (although there is a risk that one day this website and its associated files may disappear)[27].

2.3.3 Recommendations

Most of the FFRR entries on PDF provide some general recommendations on how to deal with a particular risk. For example, the threat of PDFs becoming inaccessible over time because of encryption can be reduced by formulating policies on what types of encryption are accepted; for existing collections Apache Preflight is able to detect encrypted PDFs, and in some cases it may be possible to obtain unencrypted versions from the original depositors or publishers.

[26] http://acroeng.adobe.com/wp/

[27] However the site has been archived by Internet Archive: https://web.archive.org/web/20130718100046/http://acroeng.adobe.com/wp/

3 Policy-based assessment

SCAPE Work Package 13 (Policy Representation) concerns itself with preservation policies. It considers three levels of policy – *guidance* policies, *procedure* policies and *control* policies[28]. Here, the *control* policy category represents low level policies that define the requirements for a specific collection, a specific preservation action, or for a specific designated community. It can, for example, be a technical profile that PDF documents must meet before being accepted for ingest. The role of the FFRR here is twofold:

1. It provides information on the issues and risks that are associated with the format (e.g. password protection, fonts that are not embedded), which helps in drafting the policy.
2. It provides information on how elements that are part of the policy can be "measured", and using which tools (e.g. how do we detect password protection in a PDF?).

This chapter shows how information from FFRR can be used as input for a policy-based assessment, where files are assessed against a list of technical criteria (which may include preservation risks). It also describes a general methodology for such a policy-based assessment, starting from a simple JP2 example; the approach is then illustrated further using a more elaborate example for PDF.

3.1 General methodology

The general methodology that is proposed here can be best described by starting with an example. Suppose we have a control policy that defines the requirements for JP2 images that are produced in a digitisation project. The control policy is defined by the following objectives:

1. File must be valid JP2.
2. Colour space must be defined by an ICC profile.
3. Resolution information must be stored in the 'capture resolution' box.

The JP2 page in FFRR[29] contains entries that cover all the above aspects. These entries show that each aspect can be tested using jpylyzer. In particular:

- validity can be assessed from the value of the 'isValidJP2' property[30] in the root element of jpylyzer's output;
- the presence of an ICC profile follows from the value of the 'meth' property in '/jpylyzer/properties/jp2HeaderBox/resolutionBox' [31];
- the existence of a capture resolution box follows from the presence of the 'captureResolutionBox' element in '/jpylyzer/properties/jp2HeaderBox/colourSpecificationBox' [32].

[28] http://www.scape-project.eu/deliverable/d13-1-final-version-of-policy-specification-model

[29] http://wiki.opf-labs.org/display/TR/JP2

[30] http://wiki.opf-labs.org/display/TR/Not+valid+JP2

[31] http://wiki.opf-labs.org/display/TR/Handling+of+ICC+profiles

[32] http://wiki.opf-labs.org/display/TR/Resolution+not+in+expected+header+fields

So, jpylyzer's output covers all objectives of our control policy. The next step is to translate the control policy into something machine-readable . For this we will use Schematron[33], which is a rule-based validation language that allows one to check XML trees for the presence or absence of patterns. Each objective can now be expressed as a Schematron rule:

1. File must be valid JP2:

```
<s:rule context="/jpylyzer">
  <s:assert test="isValidJP2 = 'True'">no valid JP2</s:assert>
</s:rule>
```

2. Colour space must be defined by an ICC profile:

```
<s:rule context="/jpylyzer/properties/jp2HeaderBox/resolutionBox">
  <s:assert test="captureResolutionBox">no capture resolution box</s:assert>
</s:rule>
```

3. Resolution information must be stored in the 'capture resolution' box:

```
<s:rule context="/jpylyzer/properties/jp2HeaderBox/colourSpecificationBox">
  <s:assert test="meth = 'Restricted ICC'">METH not 'Restricted ICC'</s:assert>
</s:rule>
```

Note how each rule's 'context' attribute defines the location of the element that is being evaluated in the XML (i.e. jpylyzer's output), and the 'assert' element contains the actual test (i.e. the expression that is evaluated). If the rules are combined in a schema, this allows one to asses any given JP2 against our policy using a two-step procedure:

1. run jpylyzer on the JP2;
2. validate jpylyzer's output against the schema.

The second step requires a Schematron validator. Several open-source implementations exist; examples are the ISO Schematron reference implementation[34] and Probatron[35]. The Schematron validation step results in XML output, in which each test that failed the validation is represented in a 'failed-assert' element. For example, a file that failed the 'valid JP2' objective will result in the following output:

```
<svrl:failed-assert test="isValidJP2 = 'True'" location="/jpylyzer[1]" line="45"
col="550">
  <svrl:text>no valid JP2</svrl:text>
</svrl:failed-assert>
```

Figure 3 summarises the above procedure. Although our current example is extremely simple and only covers JP2 and jpylyzer[36], the methodology is generic and applicable to any format, provided that the analysis tool is able to report its result as XML. This is shown in the next section, which presents a more complex policy-based assessment of a large collection of PDF documents.

---

[33] http://xml.ascc.net/resource/schematron/

[34] http://www.schematron.com/implementation.html

[35] http://www.probatron.org/index.html

[36] A more realistic example for JP2 is given here: http://www.openplanetsfoundation.org/blogs/2012-09-04-automated-assessment-jp2-against-technical-profile

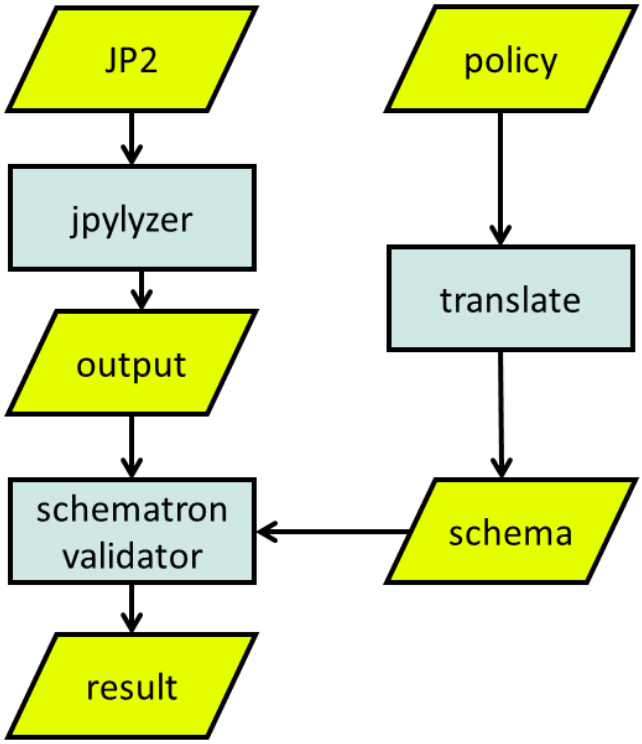

Figure 3 *Policy-based assessment of a JP2 file using jpylyzer and Schematron. The policy is manually translated to a set of Schematron rules; the resulting schema is then used to assess the output of jpylyzer for any given JP2 using a Schematron validator.*

3.2 Policy-based assessment of PDF

This second example shows how the aforementioned methodology can be used to assess a moderately large (15,000) collection of PDFs[37]. The test data were taken from Govdocs Selected[38], which is a subset of the Govdocs1 corpus[39]. An attempt was made to assess these PDFs against a (hypothetical) control policy was that is defined by the following objectives:

1. File must not be encrypted or password protected.
2. Fonts must be embedded and complete.
3. File must not contain JavaScript.
4. File must not contain embedded files (i.e. file attachments).

[37] An elaborate discussion of the work presented in this section is available here: http://www.openplanetsfoundation.org/blogs/2014-01-27-identification-pdf-preservation-risks-analysis-govdocs-selected-corpus

[38] http://www.openplanetsfoundation.org/blogs/2012-07-26-1-million-21000-reducing-govdocs-significantly

[39] http://digitalcorpora.org/corpora/files

5. File must not contain multimedia content (audio, video, 3-D objects).
6. File should be valid PDF.

Again, the PDF entries of FFRR[40] cover all these aspects; they also show how Apache Preflight reports information on these in its output. This makes it possible to translate each objective in the policy into one or more Schematron rules. As an example, below is the rule for the encryption objective:

```
<s:pattern name="Checks for encryption">
  <s:rule context="/preflight/errors/error">
    <s:assert test="not(code = '1.0' and contains(details,'password'))">Open
      password</s:assert>
    <s:assert test="not(code = '1.4.2')">Encryption</s:assert>
  </s:rule>
</s:pattern>
```

Note how the above rule comprises two separate tests (i.e. 'assert' elements). Some of the other objectives resulted in an even greater amount of tests. An extreme example of this is the 'fonts must be embedded and complete' objective, which is linked to no less than 25 separate error codes in Preflight, each of which must be represented as a separate test. After all the objectives were translated to Schematron rules, the full set of rules was combined in a schema[41]. The actual validation of the 15,000 PDFs was implemented as a simple shell script[42] that traverses a user-defined directory tree, and then performs the following actions on each PDF file inside it:

1. analyse each PDF with Apache Preflight;
2. validate Preflight's output against the schema;
3. add the outcome of the policy-based validation to a text file.

Table 1 below summarises the results of this exercise:

Outcome	Number of files	%
Pass	3973	26
Fail	11120	74

Table 1 *Outcome of policy-based assessment of PDFs in Govdocs Selected corpus.*

So, only 26% of all PDFs in Govdocs Selected meet the requirements of our control policy. Figure 4 provides more information on why this is happening. The vertical axis of the figure shows the error messages associated with the individual tests (i.e. 'assert' elements in the schema) that are part of the policy. The bars along the horizontal axis represent the percentage of analysed files for which each error was reported. The figure reveals that that the majority of failed tests are related to fonts. It may be that the policy is too strict (as it includes all font errors that are reported by Preflight). On the other hand,

[40] http://wiki.opf-labs.org/display/TR/Portable+Document+Format

[41] Available here: https://github.com/openplanets/pdfPolicyValidate/blob/master/schemas/pdf_policy_preflight_test.sch

[42] Provided here (along with additional post-processing scripts): https://github.com/openplanets/pdfPolicyValidate

font problems are known to be common in PDF, and a 2013 survey by the PDF Association showed that its members see fonts as the most challenging aspect of the format, both for processing and writing[43].

Aside from font issues, other caveats are the lack of reliable tools to test for overall conformity to the PDF specification (ISO 32000[44]), and the limited availability of reliable ground truth, which makes it difficult to assess the accuracy of the assessment results for complex documents. Within these limitations, however, the results do demonstrate the overall feasibility of a policy-based assessment of PDF documents using a combination of a PDF/A validator (in this case Apache Preflight), Schematron and information from FFRR.

3.3 Concluding remarks

In the previous sections a methodology was described for a policy-based assessment of file against technical criteria. The methodology is generic, in the sense that it can be used with any analysis tool that is able to report its output as well-formed XML. For tools that do not support XML output, additional post-processing of output to XML could be used as a workaround. In the examples shown, for each format one single tool (jpylyzer for JP2, Apache Preflight for PDF) was able to test for all the objectives that were part of the policy. There may be situations when multiple tools are needed. For example, some objectives of a policy may only be covered by tool *A*, whereas for some other objectives we may rely exclusively on tool *B*. There are a number of ways to deal with this. One approach would be to combine the XML output of multiple tools into a container file (where each tool's output is assigned its own namespace). Another possibility would be to work with the original output files, and use multiple schemas. Either way, adapting the outlined methodology for such cases would be straightforward.

[43] http://duff-johnson.com/wp-content/uploads/2014/01/PDFValidationDreamOrYawn.pdf

[44] http://acroeng.adobe.com/PDFReference/ISO32000/PDF32000-Adobe.pdf

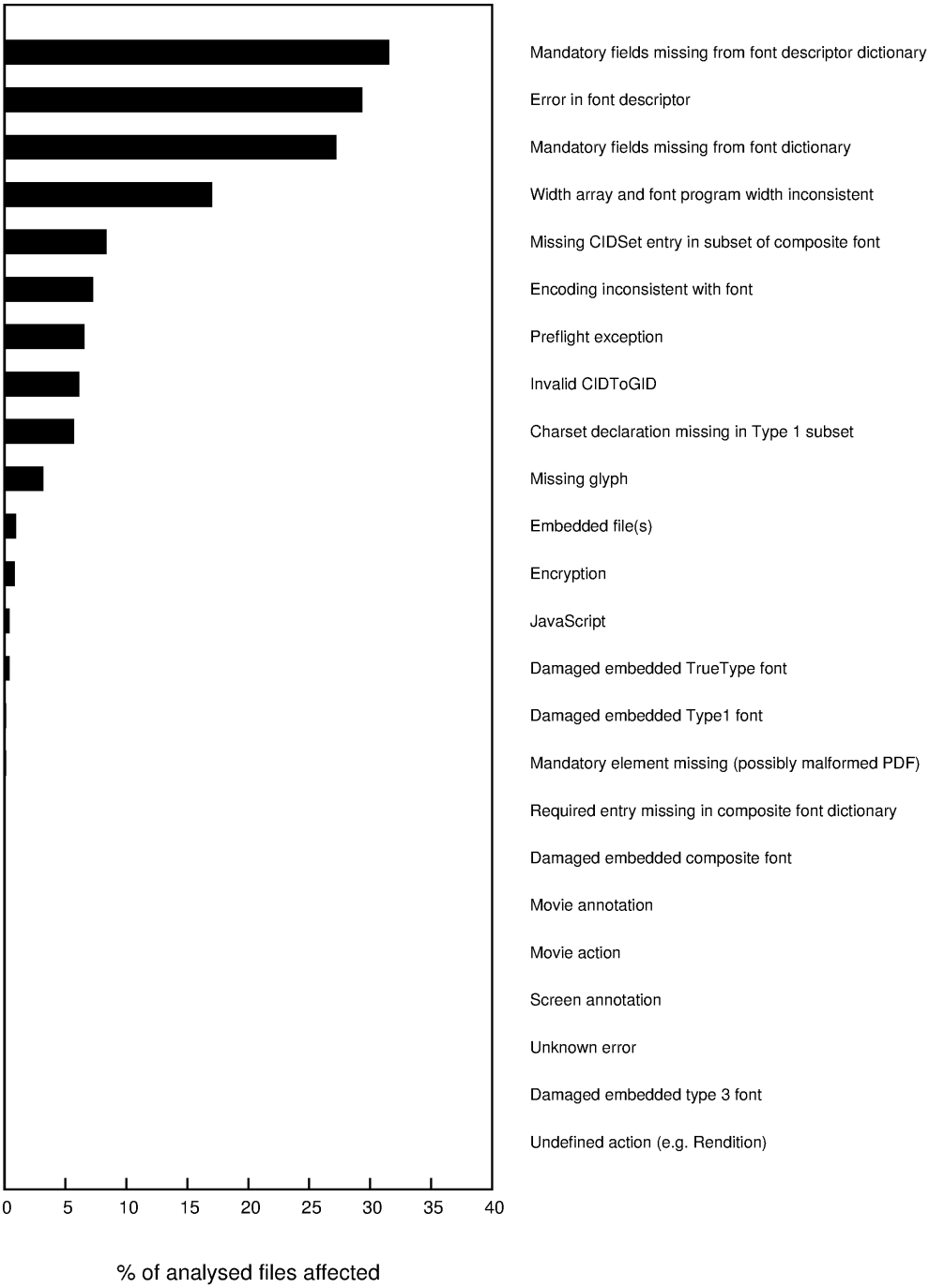

Figure 4 *Relative occurrences of failed tests in policy-based assessment of PDFs in Govdocs Selected corpus*

4 Relations with other activities

This chapter explains how the work that is part of this deliverable relates to other activities both within and outside of SCAPE. Figure 5 provides a general overview of the relations with SCAPE's preservation watch, policies and quality assurance activities, which are described in the following sections.

Figure 5 *Relation between OPF File Format Risk Registry and SCAPE preservation watch, policies and quality assurance activities (modified from Faria et al., 2014[45])*

4.1 Preservation watch and SCOUT

SCOUT is the preservation watch system that is being developed as part of Work Package 12 (Deliverable D12.2[46]). SCOUT "provides an ontological knowledge base to centralize all necessary

[45] http://www.scape-project.eu/deliverable/d12-2-final-version-of-the-preservation-watch-component

[46] Ibid.

information to detect preservation risks and opportunities"[47]. This is *not* the same thing as the "knowledge base" that is covered by this deliverable, which was given the moniker *OPF File Format Risk Registry* (FFRR) in order to avoid any confusion between them. The relation between the SCOUT knowledge base and the FFRR is illustrated by Figure 5. The SCOUT knowledge base is a central hub that makes available information that is collected from a wide variety of sources. The OPF File Format Risk Registry is just one of the many (potential) sources that may be used as input into SCOUT.

4.2 Policies

Work package 13 (Policy Representation) concerns itself with preservation policies. It considers three levels of policy – *guidance* policies, *procedure* policies and *control* policies. Here, the *control* policy category represents low level policies that define the requirements for a specific collection, a specific preservation action, or for a specific designated community[48]. It can, for example, be a technical profile that PDF documents must meet before being accepted for ingest. The role of the FFRR here is twofold:

3. It provides information on the issues and risks that are associated with the format (e.g. password protection, fonts that are not embedded), which helps in drafting the policy.
4. It provides information on how elements that are part of the policy can be "measured", and using which tools (e.g. how do we detect password protection in a PDF?).

This relation is again shown in Figure 5. Note how the policies provide another (indirect) link with the preservation watch system (SCOUT).

4.3 Quality Assurance

The FFRR also provides input to the other quality assurance activities that are part of Work Package 11[49]. It does so by first providing information on specific aspects/risks that must be accounted for in a quality assurance workflow (e.g. validity of JP2 images in an image migration workflow, or checking for password-protection in a document workflow). Moreover, it also explains how these specific aspects can be measured, and using which tool(s).

4.4 Existing format registries

A number of registries with information on file formats exist. Well-known examples are PRONOM[50], UDFR[51], Archive Team's file formats wiki[52] and the Digital Formats Web site by Library of Congress[53]. Such registries do not usually address file format risks directly, and the information in FFRR should be

[47] http://openplanets.github.io/scout/

[48] http://www.scape-project.eu/deliverable/d13-1-final-version-of-policy-specification-model

[49] http://www.scape-project.eu/deliverable/d11-2-quality-assurance-workflow-release-2-release-report

[50] http://www.nationalarchives.gov.uk/PRONOM/

[51] http://udfr.org/

[52] http://fileformats.archiveteam.org/

[53] http://www.digitalpreservation.gov/formats/

seen as complementary to them. FFRR does in fact provide links to the aforementioned registries, and FFRR itself and its associated data sets are referenced by Archive Team's wiki[54].

4.5 Uptake of results by industry and community

4.5.1 Jpylyzer

Goobi is a popular open-source workflow management software product that is widely used for digitisation projects. Intranda GmbH, the company that does most of its development, has created a JPEG 2000 validation plugin for Goobi that is based on jpylyzer[55]. Ex Libris has created a jpylyzer technical metadata extractor plugin for its Rosetta product[56]. Jpylyzer has also been adopted by the Debian/Ubuntu community, which has resulted in an improved visibility and availability on various platforms and download sites[57].

4.5.2 Apache Preflight

The initial tests with Apache Preflight revealed a number of problems with the software. These were reported back to the Apache developer community, who used this feedback to improve the software. The result of this is that Preflight has improved considerably since the first tests, both in terms of the number of problems it picks and its overall stability. The blog posts on the work with Preflight have also attracted the attention of the wider archiving community, and Preflight has since been used in a number of workshops and hackathons. For example, a 2013 hackathon of the SPRUCE project resulted in a proof-of-concept tool for the identification of PDF preservation risks, based on Preflight and FFRR[58]. SCAPE partner The British Library subsequently developed *Flint*, which is a modularised framework that offers file/format validation and policy-based assessment with a standardised xml output. Its PDF module is based on the work presented in section 3.2 of this report[59].

---

[54] See for example the "Sample files" and "Links" sections here: http://fileformats.archiveteam.org/wiki/JP2 and here: http://fileformats.archiveteam.org/wiki/PDF

[55] http://www.digiverso.com/en/products/goobi/history/42-products/goobi/329-history-goobi-1-9-2-intranda-edition

[56] https://developers.exlibrisgroup.com/blog/Jpylyzer-Technical-Metadata-Extractor-Plugin

[57] E.g. https://packages.debian.org/nl/sid/python/python-jpylyzer; https://launchpad.net/ubuntu/+source/jpylyzer

[58] http://www.openplanetsfoundation.org/blogs/2013-03-15-pdf-eh-another-hackathon-tale

[59] https://github.com/openplanets/flint


SCAPE

5 Conclusions

The OPF File Format Risk Registry (FFRR) is a platform for documenting preservation risks that are associated with file formats. Apart from describing preservation risks, it also gives information on how to assess if a file object is affected by a risk, and using what tool(s). In addition, it points to sample files that serve as supporting evidence. Finally, it provides recommendations on how institutions can deal with affected file objects. FFRR should be seen as complementary to existing format registries, which typically do not directly address preservation risks. The formats JP2 and PDF were used as a proof of concept of the FFRR approach. As the assessment of file objects for the presence of format risks requires analysis tools, research was done on the Apache Preflight tool for PDF. The results were incorporated into the FFRR entries for this format. In the absence of a suitable analysis tool for JP2, a new validation and feature extraction tool was developed: jpylyzer.

FFRR provides information on how elements that are part of a preservation policy can be measured, and using what tools. To demonstrate FFRR's role in this process, a simple methodology was developed for a policy-based assessment based on Schematron rules. The application of this methodology to a corpus of 15,000 PDFs did reveal some limitations, which were mostly due to the complexities of dealing with font-related issues in PDF, and the lack of an authoritative tool for testing overall conformance to the PDF standard. Nevertheless, it confirmed the overall feasibility of the approach.


SCAPE

Annex: Products and publications

The following is an overview of all products and publications that have been created as part of the Knowledge Base task.

OPF File Format Risk Registry

Link: http://wiki.opf-labs.org/display/TR/OPF+File+Format+Risk+Registry

jpylyzer software

Jpylyzer, JP2 validator and extractor (software). Link: http://openplanets.github.io/jpylyzer/

Jpylyzer User Manual. Link: http://openplanets.github.io/jpylyzer/userManual.html

Corpora and data

Jp2k test corpus (annotated corpus of images, primarily intended to demonstrate preservation risks in FFRR). Link: https://github.com/openplanets/format-corpus/tree/master/jp2k-test

Jpylyzer test files (annotated corpus of images, primarily intended to support jpylyzer development). Link: https://github.com/openplanets/jpylyzer-test-files

The Archivist's PDF Cabinet of Horrors (annotated corpus of PDFs, primarily intended to demonstrate preservation risks in FFRR). Link: https://github.com/openplanets/format-corpus/tree/master/pdfCabinetOfHorrors

Publications on jpylyzer and JPEG 2000

Van der Knijff, Johan, van der Ark, René & Wilson, Carl, 2012. Improved Validation and Feature Extraction for JP2 (JPEG 2000 Part 1) Images: The jpylyzer Tool. Proceedings, Archiving 2012, Copenhagen. Link: http://www.imaging.org/IST/store/epub.cfm?abstrid=45319

Tarrant, David & Johan van der Knijff, 2012. Jpylyzer: analysing JP2000 files with a community-supported tool. Proceedings, iPRES 2012, Toronto. Link: http://eprints.soton.ac.uk/341992/1/iPres2012.pdf

Van der Knijff, Johan, 2011. A simple JP2 file structure checker.
Link: http://www.openplanetsfoundation.org/blogs/2011-09-01-simple-jp2-file-structure-checker

Van der Knijff, Johan, 2011. A prototype JP2 validator and properties extractor.
Link: http://www.openplanetsfoundation.org/blogs/2011-12-14-prototype-jp2-validator-and-properties-extractor

Van der Knijff, Johan, 2012. Update on jpylyzer. Link: http://www.openplanetsfoundation.org/blogs/2012-04-23-update-jpylyzer

Van der Knijff, Johan, 2012. Automated assessment of JP2 against a technical profile.
Link: http://www.openplanetsfoundation.org/blogs/2012-09-04-automated-assessment-jp2-against-technical-profile

Van der Knijff, Johan, 2013. ICC profiles and resolution in JP2: update on 2011 D-Lib paper.
Link: http://www.openplanetsfoundation.org/blogs/2013-07-01-icc-profiles-and-resolution-jp2-update-2011-d-lib-paper

Van der Knijff, Johan, 2013. Adventures in Debian packaging.
Link: http://www.openplanetsfoundation.org/blogs/2013-04-23-adventures-debian-packaging

Publications on PDF

Van der Knijff, Johan, 2012. PDF - Inventory of long-term preservation risks.
Link: http://www.openplanetsfoundation.org/blogs/2012-07-26-pdf-inventory-long-term-preservation-risks

Van der Knijff, Johan, 2012. Identification of PDF preservation risks with Apache Preflight: a first impression (blog + report). Link: http://www.openplanetsfoundation.org/blogs/2012-12-19-identification-pdf-preservation-risks-apache-preflight-first-impression

Van der Knijff, Johan, 2013. What do we mean by "embedded" files in PDF?
Link: http://www.openplanetsfoundation.org/blogs/2013-01-09-what-do-we-mean-embedded-files-pdf

Van der Knijff, Johan, 2013. Identification of PDF preservation risks: the sequel.
Link: http://www.openplanetsfoundation.org/blogs/2013-07-25-identification-pdf-preservation-risks-sequel

Van der Knijff, Johan, 2014. Identification of PDF preservation risks: analysis of Govdocs selected corpus.
Link: http://www.openplanetsfoundation.org/blogs/2014-01-27-identification-pdf-preservation-risks-analysis-govdocs-selected-corpus

Miscellaneous publications

Van der Knijff, Johan, 2012. Magic editing and creation: a primer.
Link: http://www.openplanetsfoundation.org/blogs/2012-08-09-magic-editing-and-creation-primer

Van der Knijff, Johan, 2013. Assessing file format risks: searching for Bigfoot?
Link: http://www.openplanetsfoundation.org/blogs/2013-09-30-assessing-file-format-risks-searching-bigfoot

Van der Knijff, Johan, 2013. Measuring Bigfoot. Link: http://www.openplanetsfoundation.org/blogs/2013-10-08-measuring-bigfoot